2^{nd} assignment A.macro

Edoardo Di Cosimo, Francesco Favagrossa, Paolo Sebastiani

April 2024

I.1

In order to transform the variables in logartihm we specified the F.O.C.s using exponentials.

$$e^{y_t} = e^{z_t} \cdot (e^{k_{t-1}})^{\alpha} \tag{1}$$

$$(e^{c_t})^{-\sigma} = \beta \cdot (e^{c_{t+1}})^{-\sigma} \cdot (\alpha \cdot e^{z_{t+1}} \cdot (e^{k_t})^{\alpha-1} + (1-\delta)) \tag{2}$$

$$e^{c_t} + e^{k_t} = e^{z_t} \cdot e^{(k_{t-1})^{\alpha}} + (1-\delta) \cdot e^{k_{t-1}} \tag{3}$$

$$e^{z_t} = (1-\rho) + \rho \cdot e^{z_{t-1}} + \epsilon \tag{4}$$

I.2

We calculated the steady state values considering that now the model is expressed in exponential.

$$y_0 = \alpha \cdot k_0 \tag{5}$$

$$z_0 = \ln(1) \tag{6}$$

$$c_0 = \ln((\alpha \cdot k_0) - (\delta \cdot k_0)); \tag{7}$$

$$k_0 = \ln(((1-\beta+\beta \cdot \delta)/(\beta \cdot \alpha))/(\alpha-1)) \tag{8}$$

I.3

We generated the irfs in dynare.

I.4

We expanded the periods up to 1000, and generated the irfs again (Figure 1)

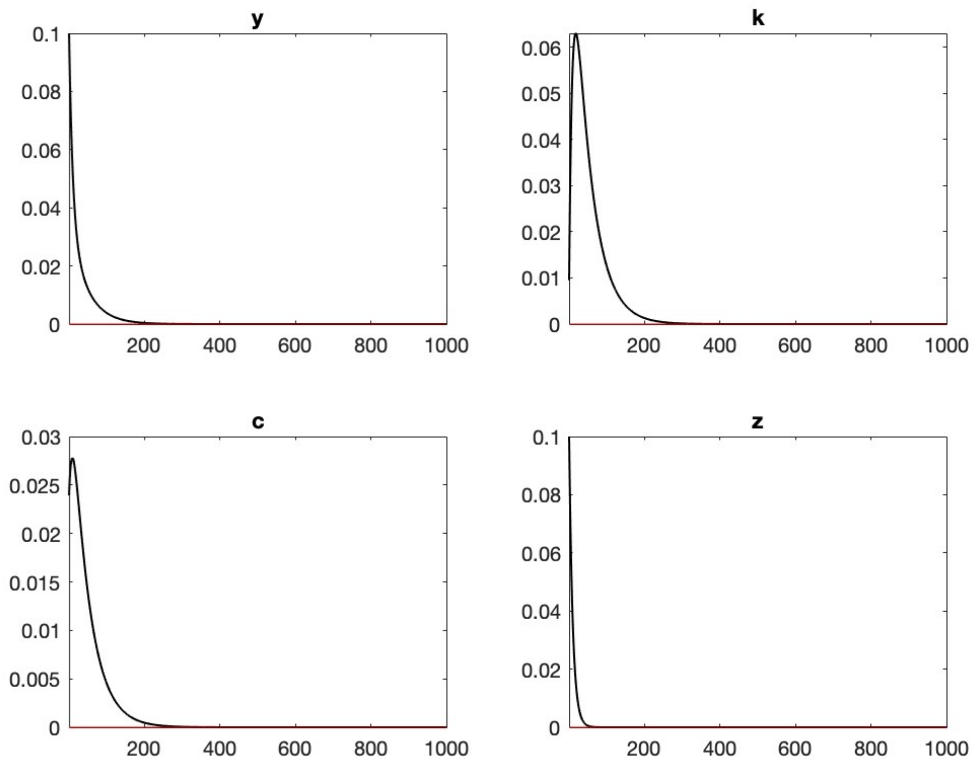

Figure 1: Plots of irfs with 1000 periods

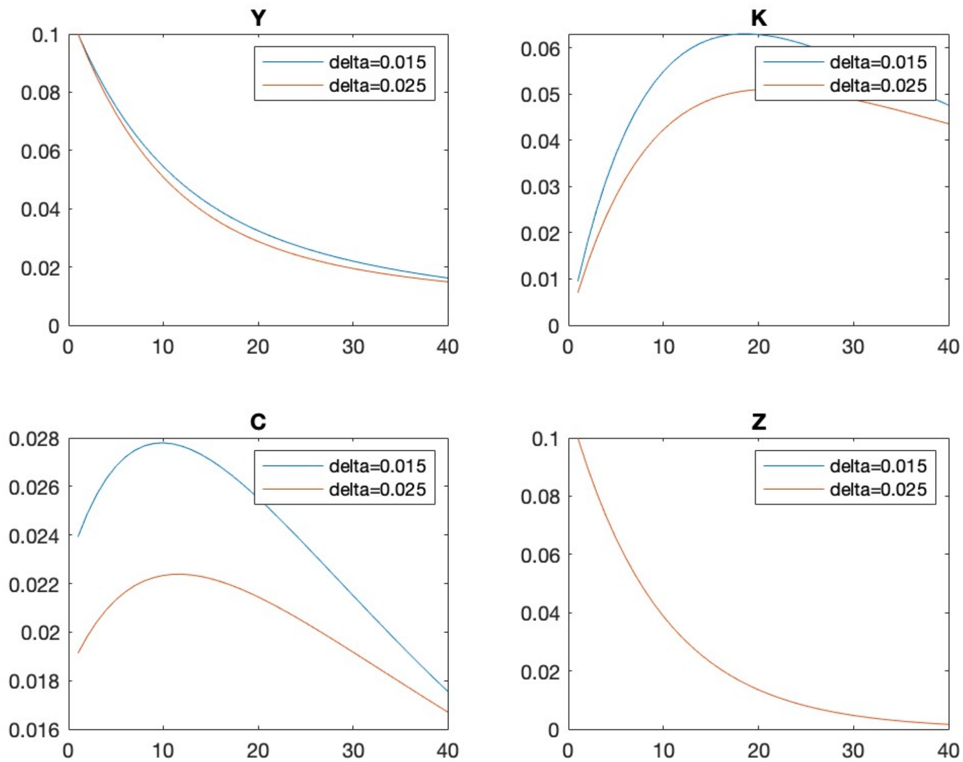

Figure 2: Plots of irfs with different δs

II.1

We created another dynare file (named del_0_015.mod) equal to the one before except for the value of delta (depreciation rate of capital) which is now equal to 0.015. We then ran both .mod files (using the noclearall command) and plotted the two different policy functions, as shown in Figure 2.

II.2

We created the matrix **X** using dynare's output.

```
X = [oo_.dr.ys';oo_.dr.ghx';oo_.dr.ghu']
```

Using the matrix we've just created we ran a *for* loop to evalute the policy function over a 40 periods timeframe. Since the results of the matlab's *for* loop are the values of the variables at each period, in order to confront them with the dynare's one (variation from steady state values) we created the vector "check" as follow:

```
yss = ones(Ti, 1) * y(1,1);
kss = ones(Ti, 1) * k(1,1);
css = ones(Ti, 1) * c(1,1);
zss = ones(Ti, 1) * z(1,1);

checkY = [y-yss,oo_.irfs.Y_eps_z'];
checkK = [k-kss,oo_.irfs.K_eps_z'];
checkC = [c-css,oo_.irfs.C_eps_z'];
checkZ = [z-zss,oo_.irfs.Z_eps_z'];
```

And we confirmed that indeed they check out.

II.3

We ran another *for* loop, but this time the shock variable is a vector of random variable with mean 0, we then plotted the irfs (Figure 3). We noticed that the variables does not converge to a value but instead they float around their steady state values. Finally we checked the moments of the simulated variables and noticed that they not check with what dynare produced (Figures 4 and 5).

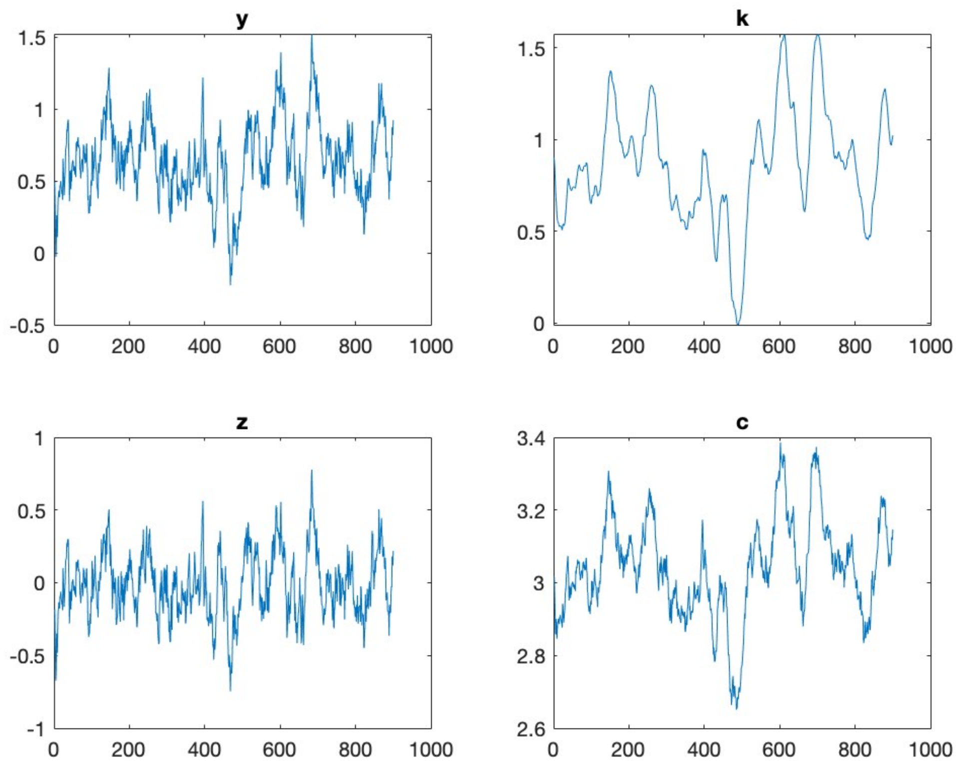

Figure 3: Simnulated irfs with random normal shocks

oo_.var

	1	2	3	4
1	0.0852	0.0811	0.0421	0.0621
2	0.0811	0.1428	0.0595	0.0384
3	0.0421	0.0595	0.0265	0.0245
4	0.0621	0.0384	0.0245	0.0519

Figure 4: Variance covariance dynare's matrix

	1	2	3	4
1	0.0699	0.0536	0.0307	0.0551
2	0.0536	0.0961	0.0398	0.0249
3	0.0307	0.0398	0.0183	0.0190
4	0.0551	0.0249	0.0190	0.0489

Figure 5: Variance covariance matrix of simulated data